Derivative–Free Optimization Methods: A Brief, Opinionated, and Incomplete Look at a Few Recent Developments

Margaret H. Wright

Computer Science Department

Courant Institute of Mathematical Sciences

New York University

Foundations of Computer-Aided Process Operations

(FOCAPO)

Savannah, Georgia

January 9, 2012

I'm delighted to be here giving this talk. Thank you for the invitation!

Disclaimer: This talk does not mention all the researchers who have worked and are working on derivative-free optimization.

For an extended, factual, and complete review of derivative-free methods and software, see "Derivative-free optimization: A review of algorithms and comparison of software implementations", Rios and Sahinidis (2011).

Optimization is a central ingredient in all fields of science, engineering, and business—notably, as all of us here know, in chemical engineering and enterprise-wide optimization.

This talk will consider the generic area of derivative-free optimization (also called non-derivative optimization), meaning unconstrained optimization of nonlinear functions using only function values.

(But we do not count methods that form explicit finite-difference approximations to derivatives.)

This talk will discuss only local optimization.

When is a derivative-free method appropriate for minimizing $f(x)$, $x \in \mathbb{R}^n$?

- Calculation of f is **very** time-consuming or expensive, even on the highest-end machines.
- f requires data collected from the real world (which may take hours, weeks, . . .).
- f is calculated through black-box software.
- f is unpredictably non-nice (e.g., undefined at certain points, discontinuous, non-smooth).
- f is "noisy" (not a precise term).

For such problems, first derivatives are often difficult, expensive, or impossible to obtain, even using the most advanced automatic differentiation.

A few examples (not from chemical engineering):

- Drug selection during cancer chemotherapy, based on the patient's measured responses, e.g. blood tests.
- Importance sampling in image synthesis.
- Automatic detection of airborne contaminants.
- Optimizing a sheet-metal press line in the automotive industry.
- Prediction of *in vitro* liver bile duct excretion.

Typically we think of two broad classes of non-derivative methods:

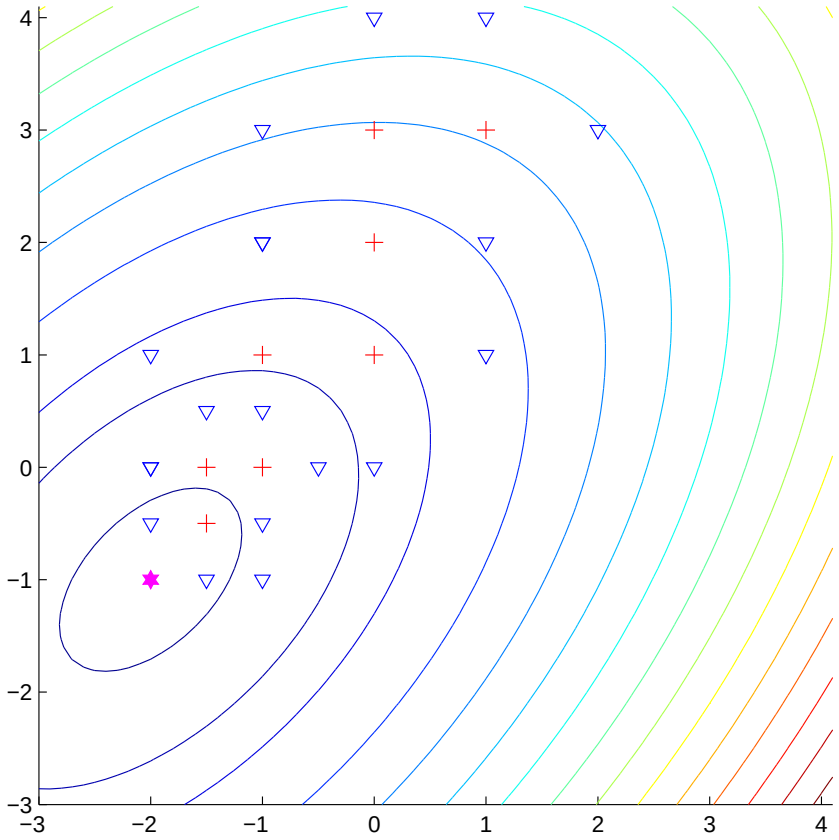
1. **"Direct search"**
   - No *explicit* model of f.
2. **Model-based**
   - Create a model of f, usually linear or quadratic, based on interpolation or least-squares, and minimize the model in some form of trust region.

NB: Genetic and evolutionary algorithms will not be considered.

A sketchy history of direct search methods (more details in Rios and Sahinidis):

- Started in 1950s (or before)—Fermi and Metropolis applied coordinate search in a 1952 paper.
- LOVED by practitioners from day 1, especially the "simplex" method of Nelder and Mead (1965).
- A fall from favor in mainstream optimization (but not with practitioners) throughout the 1970s and early 1980s.
- A renaissance starting with Torczon's (1989) PhD thesis, which gave a convergence proof for a new class (*pattern search*) of direct search methods.
- Major activity ever since, especially theoretical analysis. Too many names to list them all here!

Example: "Opportunistic" coordinate search (Fermi and Metropolis) looks for a *new best point* by taking steps along the $\pm$ coordinate directions, reducing the step if no strictly better point is found.



We now have convergence proofs (with varying definitions of "convergence", e.g. results involving $\liminf$ rather than $\lim$) for these direct search methods and more:

- pattern search and generalized pattern search,
- generating set search,
- mesh-adaptive direct search,
- frame-based and grid-restrained methods.

The proofs often require strong assumptions about f, such as twice-continuous differentiability, etc.

Nelder–Mead, to be mentioned only in passing today, is a far outlier (with extremely limited theoretical results), but nonetheless near and dear to my heart.

Typical proofs for direct search methods involve

- Requirements that the set of search directions remain "nice" (think of the coordinate directions), and
- Carefully specified acceptance criteria, e.g. simple or sufficient decrease.

The proof techniques are very closely related to those used in derivative-based optimization.

See papers by Abramson, Audet, Dennis, Kolda, Lewis, Torczon, . . .

For approximately the past 10 years, there has been major interest in model-based methods. Why?

> *If f is smooth, algorithms for unconstrained optimization are hardly ever efficient unless attention is given to the curvature of f.* (Powell).

By definition, vanilla direct search methods cannot do this.

Proofs for model-based methods need to enforce conditions on the geometry of the sample points used to define the local models. (Some very recent methods use randomness to obtain the needed conditions with high probability.)

See *Introduction to Derivative-Free Optimization*, by Conn, Scheinberg, and Vicente, SIAM (2009).

Selected recent development 1:

Within the past few years, some researchers in nonlinear optimization have become preoccupied with the *computational complexity* of many classes of unconstrained optimization methods, including Newton's method, quasi-Newton trust region and line search methods, and steepest descent.

There have been surprising results, such as that both steepest descent and Newton's method may require $O(1/\epsilon^2)$ iterations and function evaluations to drive the norm of the gradient below ϵ, even when f is bounded below and twice-continuously differentiable (Cartis, Gould, Toint, 2009).

Remember the 2-d Rosenbrock "banana" function?

$$f_R = (x_1 - 1)^2 + 100(x_2 - x_1^2)^2,$$

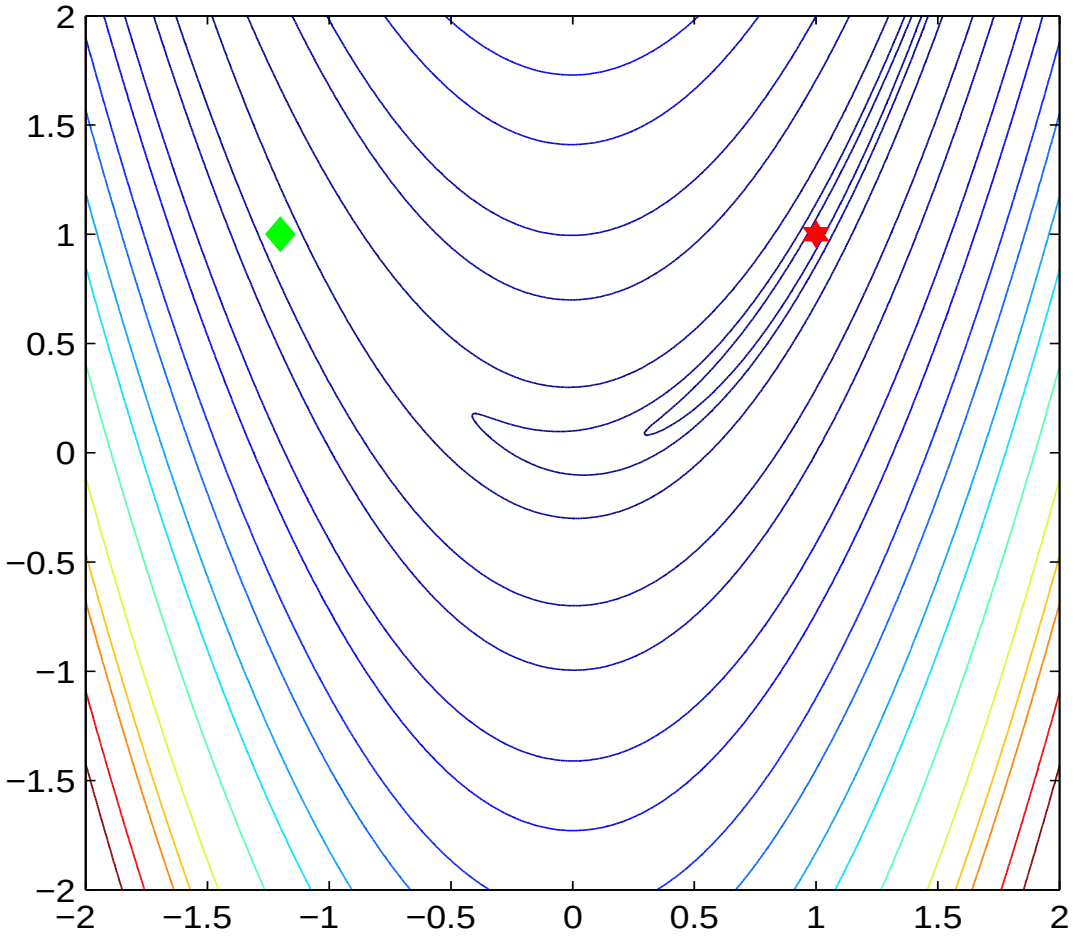
with a steep curving valley that approximately follows the parabolic path $x_2 = x_1^2$.

The unique minimizer is $x^* = (1, 1)^T$.

The "classical" starting point is $(-1.2, 1)^T$.

The difficulty of minimizing this function for descent methods is the need (unless a lucky step is taken by chance) to move around the corner.

Classical 2-d Rosenbrock



Similar figures are featured in the latest analyses of the worst-case work needed by descent methods, using a "cute" (actually, very nasty) class of functions devised by Nesterov, called the smooth Chebyshev-Rosenbrock function.

For $x \in \mathcal{R}^n$ and $\rho > 0$:

$$f_{\mathrm{CR}}(x) = \frac{(x_1 - 1)^2}{4} + \rho \sum_{i=1}^{n-1} (x_{i+1} - 2x_i^2 + 1)^2,$$

where ρ can be interpreted as a penalty parameter.

The unique minimizer of f_{CR} is $x^* = (1, 1, \ldots, 1)^T$.

"Rosenbrock" is obvious—but why "Chebyshev"?

The "penalty" term in f_{CR} is zero when $x_{i+1} - 2x_i^2 + 1 = 0$, i.e., when

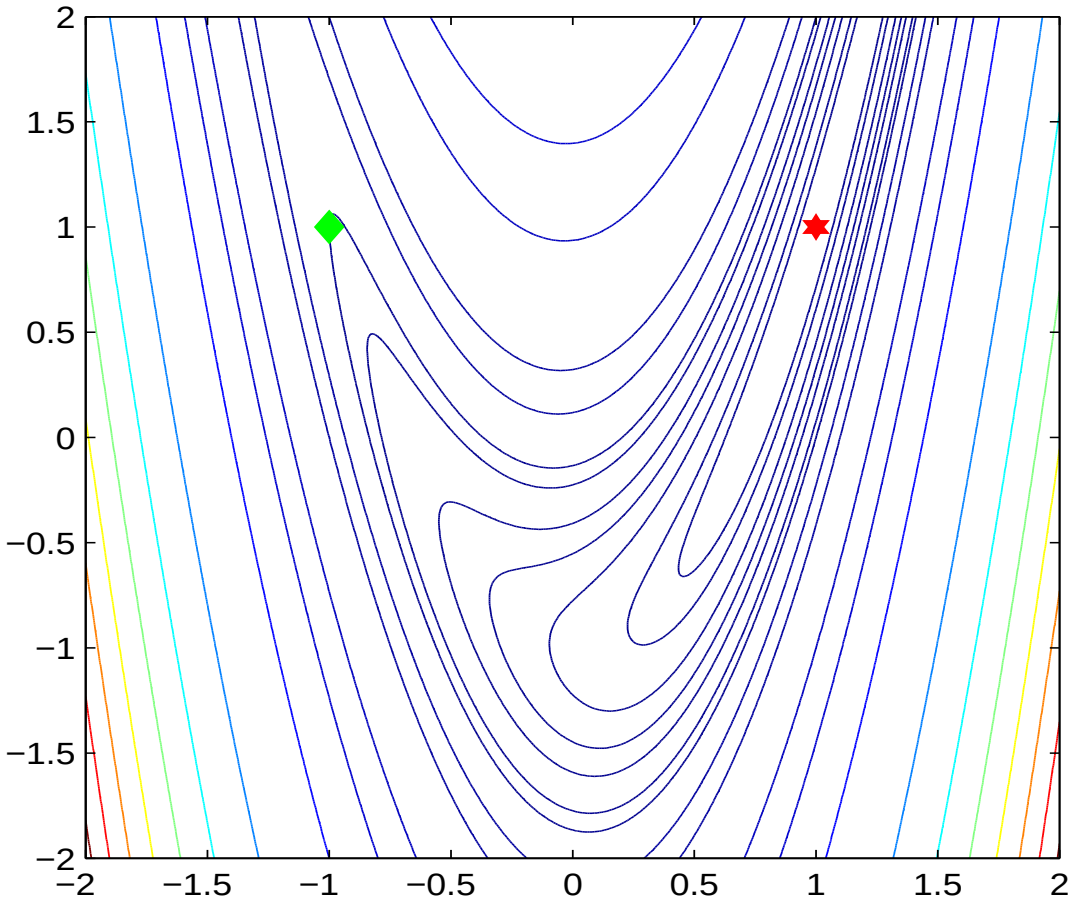
$$x_{i+1} = 2x_i^2 - 1 = T_2(x_i) = T_{2^i}(x_1),$$

where $T_k(x)$ is the kth Chebyshev polynomial of the first kind,

$$T_k(x) = \cos(k \arccos(x)), \quad \text{with} \quad T_2(x) = 2x^2 - 1.$$

Suppose we start minimizing at the very special point $x_0 = [-1, 1, 1, \ldots, 1]^T$. This point is special because $x_{i+1} - 2x_i^2 + 1 = 0$, so that $f_{\mathrm{CR}} = 1$.

f_{CR} with $\rho = 1$



Starting at x_0, using any *descent method*, in which each new iterate must have a strictly lower value of f_{CR}, iterates are forced to stay close to the manifold in which $x_{i+1} - 2x_i^2 + 1 = 0$ for $i = 1, \ldots, n-1$. This greatly restricts how far they can move.

Jarre (2011) showed that, starting at $x_0 = (-1, 1, 1, \ldots, 1)^T$, for $n \geq 12$ and $\rho \geq 400$, any continuous piecewise linear descent path from x_0 to x^* consists of at least $(1.44)(1.618^n)$ linear segments.

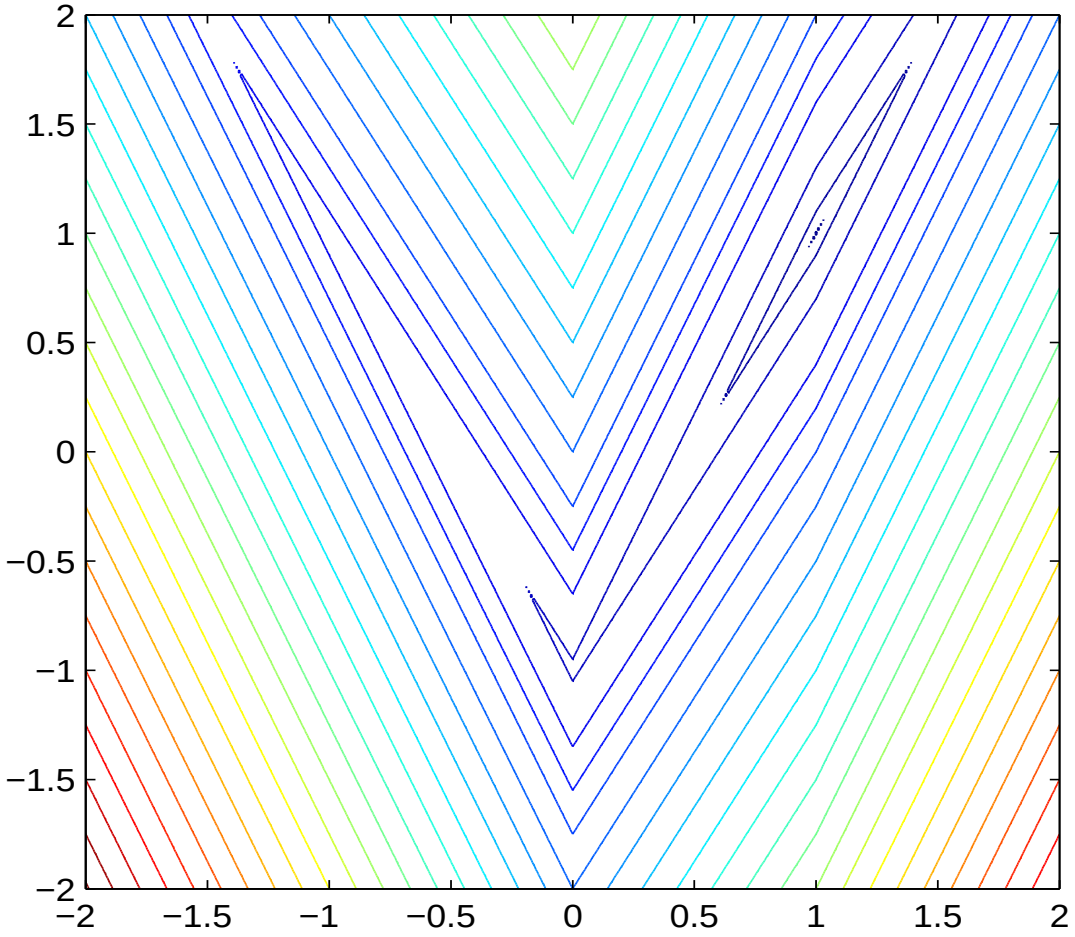
When $\rho = 400$ and $n = 12$, this means that at least 460 iterations are needed—and this is only a lower bound.

There are also two non-smooth Chebyshev-Rosenbrock functions (see Gurbuzbalaban and Overton, 2011):

$$f_{\mathrm{CR1}}(x) = \frac{(x_1 - 1)^2}{4} + \rho \sum_{i=1}^{n-1} |x_{i+1} - 2x_i^2 + 1|.$$

$$f_{\mathrm{CR2}}(x) = \frac{|x_1 - 1|}{4} + \rho \sum_{i=1}^{n-1} |x_{i+1} - 2|x_i| + 1|.$$

f_{CR2} with $\rho = 1$



The smooth and non-smooth Chebyshev-Rosenbrock functions pose severe challenges for descent derivative-free methods, even for $\rho = 1$ and small values of n.

Smooth CR, $n = 3$			
method	nfuneval	best f	best x
model-based (newuao)	1605	4.60_{-8}	$(0.9996, 0.9983, 0.9931)^T$
coordinate search	13500	1.17_{-4}	$(0.9785, 0.9136, 0.6990)^T$
Nelder–Mead**	426	5.60_{-9}	$(1.0001, 1.0006, 1.0022)^T$
Second non-smooth CR, $n = 2$			
model-based (newuao)	22	1.0	$(-1, 1)^T$
coordinate search	197	1.0	$(-1, 1)^T$
Nelder–Mead**	261	5.24_{-8}	$(1.000, 1.000)^T$

Why might these results on complexity, and these nasty functions, be interesting to practicing optimizers?

Of course they are when we're wearing our "mathematics" hats. But what are the practical implications, if any?

They show the importance of an appropriate model and the benefits of allowing non-descent steps.

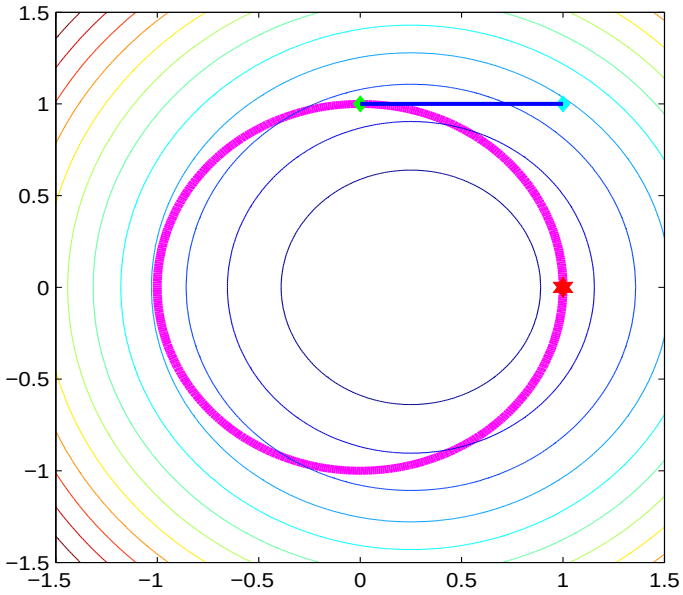
One can interpret the smooth Chebyshev-Rosenbrock function as a quadratic penalty function for the constrained problem of minimizing $(x_1 - 1)^2$ subject to the constraint $x_{i+1} - 2x_i^2 + 1 = 0$.

Because the starting point is "on" the nonlinear constraint, insisting on a strict reduction in the penalty function at each iteration forces the method to take little teeny steps.

Déjà vu!

In 1978, Maratos analyzed the lack of progress if the ℓ_1 penalty merit function is used in a constrained optimization method when an iterate lies exactly on an equality constraint but is very far from the solution.

Minimize $2(x^2 + y^2 - 1) - x$ subject to $x^2 + y^2 = 1$.



To remedy the Maratos effect, in 1982 Chamberlain and Powell developed the "watchdog" technique. The idea is to compute the step using a powerful method like Newton's method, but not to enforce descent for a (small) number of iterations. If descent has not occurred at the end of this sequence of iterations, a "fail-safe" procedure is used.

For at least 20 years, "non-monotone" versions of standard unconstrained methods have been proposed, with the idea of improving efficiency.

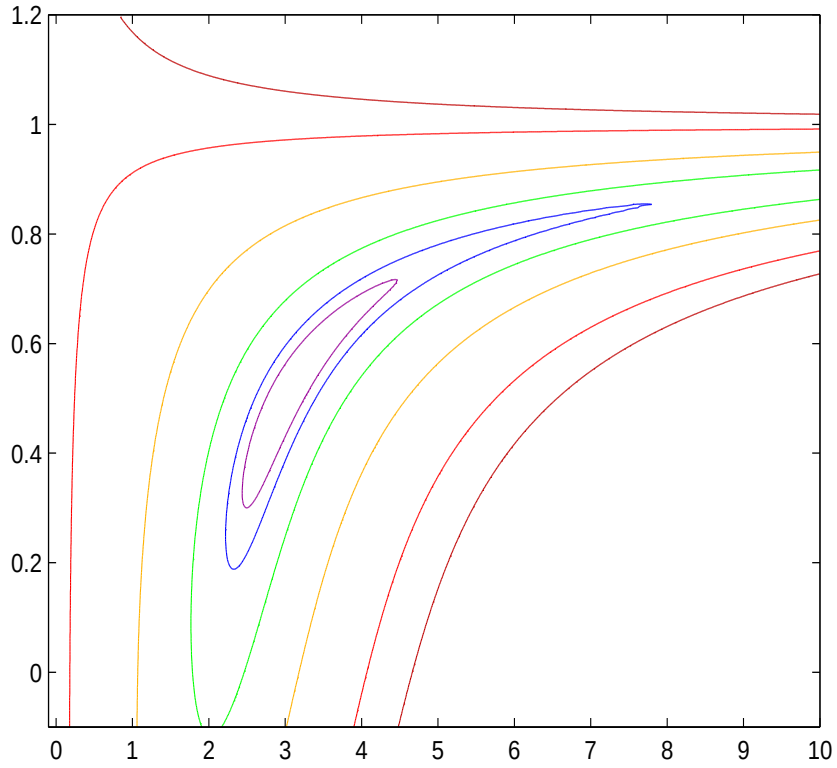
The Chebyshev-Rosenbrock examples provide a motivation for designing non-monotone derivative-free methods. . . and they also emphasize a great truth that users sometimes neglect to consider: the implications of the starting point for algorithmic performance.

Selected recent development 2:

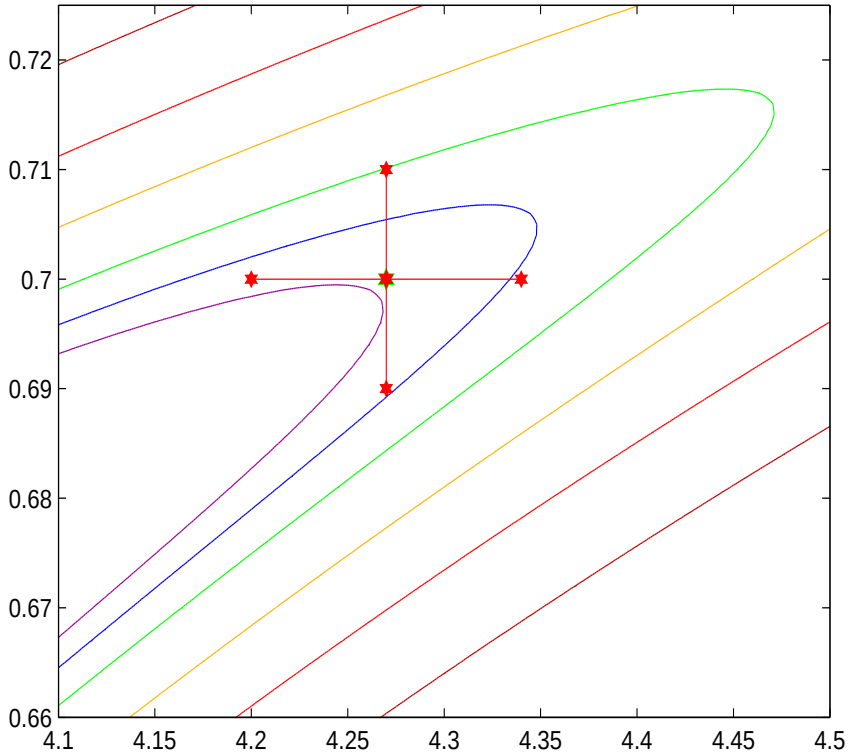
As seen on the smooth Chebyshev-Rosen problem (and as known for more than 40 years), derivative-free descent methods—especially direct search methods—can "stagnate" far from the solution, i.e. take many (hundreds, even thousands) of tiny steps without noticeably improving the function value.

Wilde (1964) noted the danger of stagnation in a "resolution valley", attributable in most cases to numerical errors or noise in evaluating f—difficulties that are still with us!

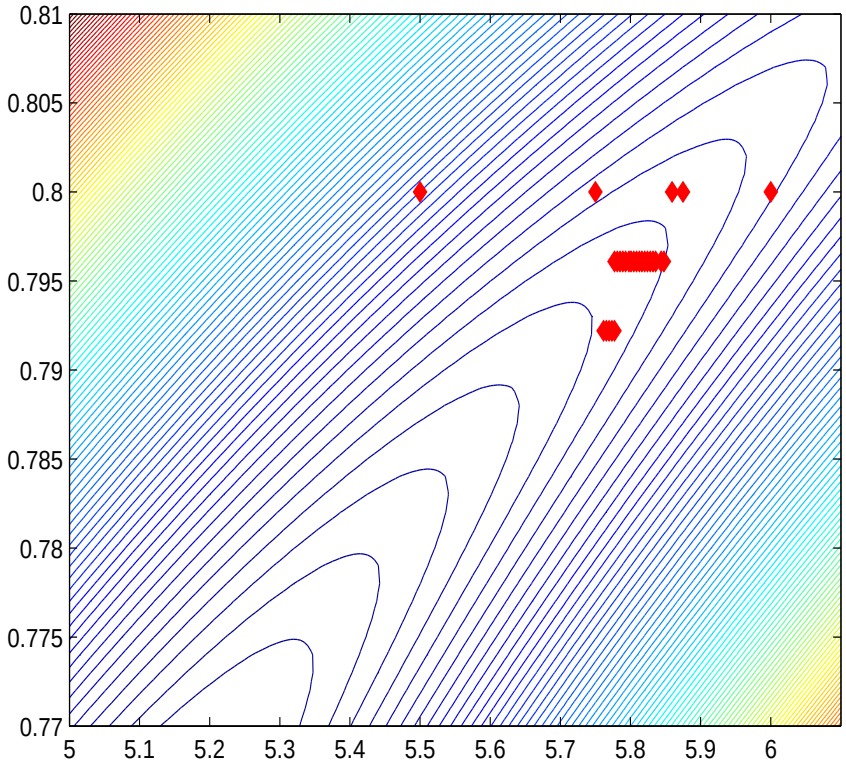
An example where some direct search methods get stuck: Beale's "awkward function", featuring curving valleys (like Rosenbrock, like Chebyshev–Rosenbrock):



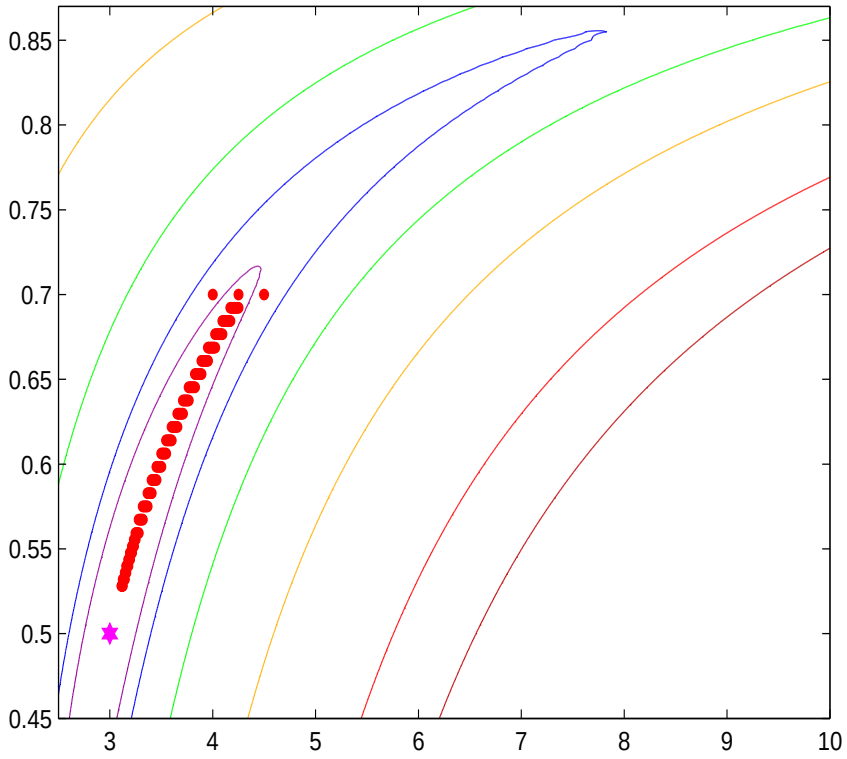
A resolution valley in Beale's function.



Here is what can happen when coordinate search is used on Beale's function:



Starting closer to the solution helps for a while, but stagnation occurs again.



How to escape from stagnation in direct search methods?

Idea: re-visit ideas from Shah, Buehler, and Kempthorne (1964), Powell (1964), and Brent (1973), where the crucial strategy is

Start by doing something random

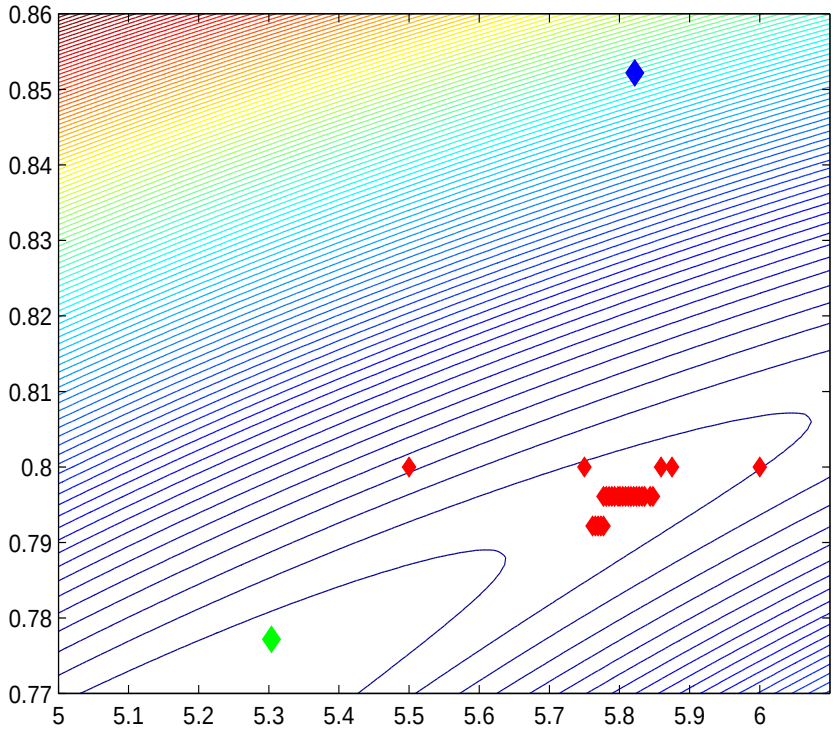
and then follow up.

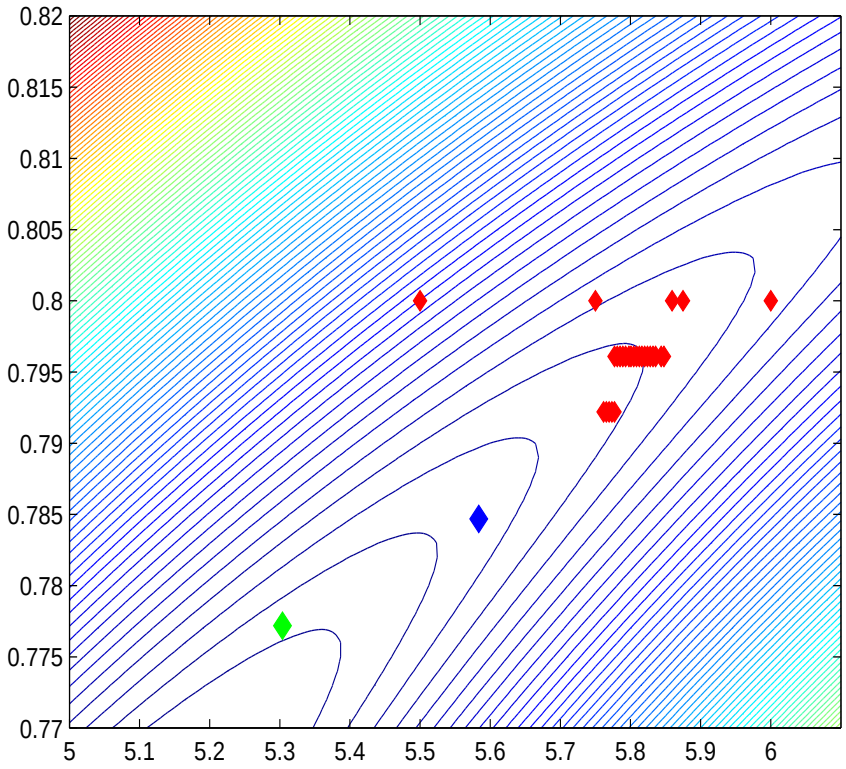
Under investigation: let $\bar{x}$ be the best point in an unproductive sequence of iterations:

1. Generate a perturbed point x_R from $\bar{x}$ by adding random multiples of the columns of a matrix V with linearly independent columns.
2. Perform line searches along the columns of V from x_R, producing the point $\hat{x}$.
3. Define a new point $\tilde{x}$ as the best point found in a line search between $\bar{x}$ and $\hat{x}$.

Because of randomness, every example is different, but the strategy seems promising. (Only 2-d results are shown.)

The blue diamond is the random point and the green diamond is the improved point, far from where the method stagnated.




0.82
0.815
0.81
0.805
0.8
0.795
0.79
0.785
0.78
0.775
0.77
5
5.1
5.2
5.3
5.4
5.5
5.6
5.7
5.8
5.9
6

Issues to be decided (knowing that there is no all-purpose answer):

1. How can the algorithm numerically identify an "unproductive" sequence of iterations?
2. Should the random multiples taken of the columns of V be related to the termination tolerances, significantly larger, or. . . ?
3. In the line searches, is a single parabolic fit enough, or is a more accurate line search needed, keeping in mind that evaluating f is assumed to be expensive?

Also see "Non-intrusive termination of noisy optimization", Larson and Wild, 2011.

Selected recent development 3:

Time for only a brief mention!

The goal is to develop model-based methods using techniques from compressed sensing and sparse approximation.

A very hot topic, already arguably overcrowded with eager researchers—23,000,000 Google hits for "compressed sensing" on January 7, 2012.

In compressed sensing, under various conditions it is possible to obtain an exact representation of an n-component sparse signal from many fewer than n measurements.

What are possible applications in non-derivative methods?

One such connection: Bandeira, Scheinberg, and Vicente (2011) have proposed a model-based method for functions with sparse Hessians in which the components of the Hessian are estimated via a linear programming subproblem at each iteration. Numerical results are very favorable, and there are proofs of convergence "with high probability".

A closely related idea, just beginning to be explored, is to use analogous techniques to develop Hessian approximations for multiscale problems in which the elements of the Hessian differ drastically in magnitude (although the Hessian is not strictly sparse).

Summary of three recent developments in derivative-free methods:

1. Analysis of the worst-case work needed for general unconstrained optimization, with implications for beneficial features in future non-derivative methods;
2. Strategies for using randomness to escape stagnation and terminate gracefully without wasting function evaluations;
3. Application of compressed sensing ideas within derivative-free methods to enhance performance on multiscale problems.

A lot for all of us to do!!!